Nr

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/810,191	03/19/2001	Masato Mitsumori	ASA-990	6919

24956 7590 03/30/2004

MATTINGLY, STANGER & MALUR, P.C.
1800 DIAGONAL ROAD
SUITE 370
ALEXANDRIA, VA 22314

EXAMINER	
RUTTEN, JAMES D	
ART UNIT	PAPER NUMBER
2122	4

DATE MAILED: 03/30/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

h

Office Action Summary	Application No.	Applicant(s)
	09/810,191	MITSUMORI ET AL.
	Examiner	Art Unit
	J. Derek Rutten	2122

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *19 March 2001*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-19* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-19* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *19 March 2001* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date 2.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-19 have been examined.

Priority

2. Acknowledgment is made of applicant's claim for foreign priority based on an application filed in Japan on 25 October 2000. It is noted, however, that applicant has not filed a certified copy of the 2000-332110 application as required by 35 U.S.C. 119(b).

Specification

3. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

The following title is suggested: "Compile method for storing source code within object code".

Claim Objections

4. Claim 2 is objected to because of the following informalities: A typo in line 12 includes the word "and", which should be removed. Appropriate correction is required.

5. Claim 17 is objected to because of the following informalities: A typo in line resulting in the word "compileraccording" which should be --compiler according--.

Claim Rejections - 35 USC § 112

6. The following is a quotation of the second paragraph of 35 U.S.C. 112:

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

7. Claims 1-19 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

8. The claims have numerous grammatical and lack of antecedent basis errors, some examples of which are recited below. This list is not intended to be an exhaustive review of all errors in the claims. For the purpose of further examination, interpretation will be made accordingly as pointed out in the subsequent prior art rejections.

9. Claim 1 recites the limitations "an object program" and "a source program" in lines 1 and 2 in the preamble and in lines 4 and 5 in the body of the claim. Further, lines 7 and 8 refer to "said object program" and "said source program", but it is not clear as to which instance this refers. For the purpose of further examination, the instances in the preamble have not been referred to in interpreting the claim. Instead, the first instance in the body has been referred to. It is noted that this presents further issues in subsequent dependent claims. In the interest of compact prosecution, interpretation is made in each subsequent claim based upon the specification.

10. Claim 2 recites the limitations "said source program", "said object program", "said input source program", and "said object program file". Due to multiple instances of each limitation (e.g. "a source program" found both in the preamble and in the body of claim 1), it is not clear as to which each particular instance refers. Further, some of the method steps in this claim occur prior to the method steps of parent claim 1, but use elements not produced until subsequent steps of claim 1 (e.g. "said object program" found in lines 5 and 6 of claim 2). For the purpose of

further examination, interpretation of claim 2 will be made in light of the above mentioned interpretations of claim 1.

11. Claim 3 recites the limitation "said object program file" in lines 7-8. There is insufficient antecedent basis for this limitation in the claim. For the purpose of further examination, this limitation has been interpreted as --an object program file--.

12. Claim 3 recites the limitations "an object program" and "a source program" in lines 1 and 2 in the preamble and in lines 5 and 6 in the body of the claim. Further, line 10 refers to "said object program", but it is not clear as to which instance this refers. For the purpose of further examination, the instances in the preamble have not been referred to in interpreting the claim. Instead, the first instance in the body will be referred to as "a first object program". Also, for the purpose of clarity, the instance of "a source program" in line 5 will be interpreted as --a first source program--. These interpretations will be used throughout all depending claims.

Claim Rejections - 35 USC § 101

13. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

14. Claims 16-19 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

Claims 16-19 merely claimed as a "compiler", that is mere arrangements or compilations of facts, information, or data structure *per se*, and/or program listing, without creating any functional interrelationship, either as part of the stored data or as part of the computing processes performed by the computer ("acts"), then such descriptive material alone does not impart functionality either to the data as so structured, or to the computer. Thus, such "descriptive material", non-functional descriptive material, that cannot exhibit any functional interrelationship with the way in which computing processes are performed does not constitute a statutory process, machine, manufacture or composition of matter. And the purely non-functional descriptive material cannot alone provide the practical application for the manufacture. **Warmerdam**, 33 F.3d at 1361, 31 USPQ2d at 1760. **In re Sarkar**, 588 F.2d 1330, 1333, 200 USPQ 132, 137 (CCPA 1978). See Examination Guidelines for Computer-Related Inventions-Final Version, pages 9 & 10. See MPEP § 2106(IV)(B)(1)(b).

15. Given 35 USC 112 deficiency issues as noted above, and in the interest of compact prosecution, claims 1-19 are best interpreted by the examiner based on the broadest reasonable interpretation of the language explicitly recited in the claims, and are further rejected as set forth below in view of the prior art.

Claim Rejections - 35 USC § 102

16. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

17. Claims 1, 3, 9, and 10 are rejected under 35 U.S.C. 102(b) as being anticipated by U.S. Patent 5,287,548 to Flood et al (hereinafter referred to as "Flood").

As per claim 1, Flood discloses:

In a compiler for generating an object program from a source program, a compile method comprising the steps of:

storing a first object program, generated by compiling a first source program, in a first object program file (column 2 line 66 – column 3 line 5: "A programmable controller system according to the present invention employs **a compiler** to generate machine language object code instructions from a source code control program written in a higher level language. A storage device is included in the controller which **retains the object code** program along **with those portions of the source code** which can not be easily reconstructed from the corresponding object code."); *and*

storing said first source program corresponding to said first object program in said first object program file, said first source program being associated with said first object program in said first object program file (column 3 lines 2-5 as cited above.).

As per claim 3, Flood discloses:

In a compiler for generating an object program from a source program, a compile method comprising the steps of:

obtaining first source information by analyzing syntax of a first source program input to said compiler (column 2 lines 66-68 as cited in the above rejection of claim 1 discloses a compiler. Compilers inherently obtain source information by analyzing source program input, otherwise they would be unable to generate the required object code.);

All other limitations have been addressed in the above rejection of claim 1.

As per claim 9, Flood discloses *a computer-readable recording medium having recorded therein a program for executing the compile method according to claim 1* (FIG. 1 element 24).

As per claim 10, Flood discloses *a computer-readable recording medium having recorded thereon a program generated by the compile method according to claim 1* (FIG. 1 element 10).

Claim Rejections - 35 USC § 103

18. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

19. Claim 2 is rejected under 35 U.S.C. 103(a) as being unpatentable over Flood as applied to claim 1 above, and further in view of Japanese Patent JP402201542A to Ota et al. (hereinafter referred to as "Ota").

As per claim 2, Flood does not expressly disclose comparing source code prior to compilation.

However, in an analogous environment, Ota teaches:

the steps of:

before generating an object program, comparing [[said]] a second source program input to the compiler with a source program corresponding to said first object program stored in an object program file;

detecting a changed portion in said second source program since the previous compiling (Abstract, paragraph 2: "`A compiler 1 compares a source program 2 of the latest generation and a source program of the preceding generation with each other to pick up correction positions.`");

compiling the changed portion of said second input source program; and

storing a second object program, generated by compiling the changed portion of said second input source program, in said first object program file (Abstract paragraph 2: "`Thereafter, an object program 5 of the preceding generation is inputted and is corrected in accordance with correction positions of the source program to output an object program`

4 of the latest generation." <u>Comment</u>: Compilation of the changed portion is inherent in the correction and output of "object program 4 of the latest generation.").

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Ota's source comparison method with Flood's source/object storage. One of ordinary skill would have been motivated to reduce compilation time by only compiling the changed portions of code.

20. Claim 4 is rejected under 35 U.S.C. 103(a) as being unpatentable over Flood as applied to claim 3 above, and further in view of "Kernel Korner: The ELF Object File Format by Dissection" by Youngdale (hereinafter referred to as "Youngdale").

As per claim 4, Flood discloses:

storing said second object program and the changed portion of said second source information in said first object program file (Flood column 14 lines 25-34).

Flood does not expressly disclose comparing source code prior to compilation.

However, in an analogous, Ota teaches:

the steps of:

when generating <u>a second</u> [said] object program, comparing <u>a second</u> source program input to said compiler with <u>said first</u> source program stored in <u>said first</u> object program file; detecting a change in said second source program since previous compiling (Abstract, paragraph 2: "A compiler 1 compares a source program 2 of the latest generation and a source program of the preceding generation with each other to pick up correction positions.");

compiling the changed portion of said second source information of the second input source program (Abstract paragraph 2)*; and*

The combination of Flood and Ota does not teach the comparison of analysis information.

However, in an analogous environment, Youngdale further teaches:

source information obtained by analyzing syntax of a second source program (bottom of page 1: "Each ELF file contains a table that describes the sections within the file.").

It would have been obvious to one of ordinary skill in the art at the time the invention was made to compile Flood's code using Ota's comparison of Youngdale's source information. One of ordinary skill would have been motivated to examine object code information pertaining to various indirect factors regarding code generation such as code section location, layout, machine, or version information to provide consistent quality object code in the context of a dynamic software development environment.

21. Claim 5 rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Flood and Ota as applied to claim 2 above, and further in view of U.S. Patent 5,586,328 to Caron et al. (hereinafter referred to as "Caron").

As per claim 5, the combination of Flood and Ota discloses:

compiling said changed portion in said source program; and storing said source program in said object program file, said source program or said source information being associated with said object program in said object program file. (as cited in the rejections of claims 1 and 2).

The combination of Flood and Ota does not expressly disclose comparison on a procedural basis.

However, in an analogous environment, Caron teaches:

comparing said source programs on a procedure basis of said source program (column 3 lines 17-20)*; and*

storing said source program on a procedure unit basis (column 3 lines 8-12).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the source code compilation methods of Flood and Ota with the unit compilation of Caron. One of ordinary skill would have been motivated to maintain the integrity of an object file, which depends on correct ordering and reference of data within blocks.

22. Claim 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Flood and Ota as applied to claim 2 above, and further in view of U.S. Patent 5,170,465 to McKeeman et al. (hereinafter referred to as "McKeeman").

As per claim 6, the combination of Flood and Ota discloses:

A compile method according to claim 2, further comprising the steps of:

comparing said source programs; compiling said changed portion in said source program; and storing said source program in said object program file, said source program or said source information being associated with said object program in said object program file. (These limitations have been addressed in the above rejection of claim 2)

The combination of Flood and Ota does not expressly disclose comparison on a processing step basis.

However, in an analogous environment, McKeeman teaches:

comparing on a processing step unit basis (column 3 lines 24-30)*; and*

storing said source program on a processing step unit basis (column 3 lines 64-68).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use the source code compilation methods of Flood and Ota with the unit compilation of McKeeman. One of ordinary skill would have been motivated to provide and maintain the integrity of an object file, which depends on proper dependency and correct ordering and reference of data within blocks.

23. Claims 7 and 8 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Flood and Ota as applied to claim 2 above, and further in view of "Learning the Korn Shell" by Rosenblatt (hereinafter referred to as "Rosenblatt").

As per claim 7, the combination of Flood and Ota does not expressly disclose:

inputting an instruction specifying whether or not to compare said source program input to the compiler with said source program stored and associated with said object program in said object program file; and if an instruction not to compare is input, compiling the entire said source program input to the compiler as said changed portion.

However, in an analogous environment, Rosenblatt teaches the use of command line options to control the behavior of programs (Section 6.1).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Rosenblatt's command line options to disable the behavior of Ota's differential compiler. One of ordinary skill would have been motivated to require the compiler to recompile the entire source program as a way to debug or verify the proper operation of the system.

As per claim 8, the combination of Flood and Ota does not expressly disclose:

inputting an instruction specifying whether or not to store said source program in said object program file; and if an instruction not to store is input, storing said object program in said object program file.

However, in an analogous environment, Rosenblatt teaches the use of command line options to control the behavior of programs (Section 6.1).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Rosenblatt's command line options to disable the behavior of Flood's compiler. One of ordinary skill would have been motivated to require the compiler to recompile the entire source program as a way to debug or verify consistency and the proper operation of the system.

24. Claims 11 and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Flood in view of Caron.

As per claim 11, Flood discloses:

in said object program file, storing said plurality of object-program compile units and said plurality of source-program compile units respectively associated with one another in said object program file, said plurality of source-program compile units being used to update said object program file on an object-program compile unit basis (Flood column 3 lines 2-5 as cited in the above rejection of claim 1. In this instance, a compile unit comprises an entire program.).

Flood does not expressly disclose compilation on a procedure basis.

However, in an analogous environment, Caron teaches:

by regarding procedures of said source program as source-program compile units, compiling said source program on a procedure basis to generate said plurality of object-program compile units (Caron column 3 lines 9-13: "`While` **`compiling`** `a program, the compiler creates an import table for` **`each constituent unit`** `of the program in which it records data relating to dependencies of the unit's source code on that of another unit, and the types of the dependencies.`"; <u>Comment</u>:

Generation of object code is inherent in the compilation procedure, otherwise code would simply be analyzed with no resulting code transformations.).

It would have been obvious to one of ordinary skill in the art at the time the invention was made use Flood's compiler to compile Caron's procedures. One of ordinary skill would have been motivated to provide and maintain the integrity of an object file, which depends on proper dependency and correct ordering and reference of data within blocks.

As per claim 16, all limitations have been addressed in the above rejection of claim 11.

25. Claims 12 and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Flood in view of Caron as applied to claim 11 above, and further in view of Ota.

As per claim 12, the above rejection of claim 11 is incorporated. Flood further discloses:

after making a change in said source program, inputting a source-program compile unit of a changed source program (Flood column 14 lines 25-34);

in said object program file, updating said read source-program compile unit so as to be the same as said input source-program compile unit and updating the stored object-program compile unit corresponding to said read source-program compile unit to said new object-program compile unit (Flood column 14 lines 25-34).

The combination of Flood and Caron does not expressly disclose comparison of compile units.

However, in an analogous environment, Ota teaches:

out of [[said]] a plurality of source-program compile units stored in said object program file, reading a source-program compile unit corresponding to said input source-program compile unit; comparing said input source-program compile unit with said read source-program compile unit (Ota Abstract)*;*

if [[the two of]] said input source-program compile unit and said read source-program compile unit do not coincide, compiling said input source-program compile unit to generate a new object-program compile unit (Ota Abstract)*;*

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Ota's source comparison method with the combination of Flood and Caron's procedural compilation and source/object storage. One of ordinary skill would have been motivated to reduce compilation time by only compiling the changed portions of code.

As per claim 17, all limitations have been addressed in the above rejection of claim 12.

26. Claim 13 is rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Flood, Caron and Ota as applied to claim 12 above, and further in view of prior art of record Japanese Patent Publication number 08-087416 to Satoshi (hereinafter referred to as "Satoshi").

As per claim 13, the combination of Flood, and Ota does not disclose restriction of the compilation if source code function is identical.

However, in an analogous environment, Caron teaches the syntactical and semantic analysis of source code (column 1 lines 29-40; also column 3 lines 40-44). The process of syntactical and semantic analysis translates source code into a compact object code which does not contain the formatting common in source code such as comment lines, tabs, and other "white space." As such two files that differ only by the content of comments or white space are otherwise equivalent to a compiler that performs syntax analysis.

Caron does not expressly teach suppression of a compilation process.

However, in an analogous environment, Satoshi teaches inhibiting a compilation when source files appear to be equivalent (Abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Caron's syntax analysis in Flood's compiler with Satoshi's compiler interruption method. One of ordinary skill would have been motivated to speed up compilation time by avoiding the compilation of semantically identical source files.

27. Claims 14, 15, 18, and 19 are rejected under 35 U.S.C. 103(a) as being unpatentable over the combination of Flood and Caron as applied to claim 11 above, and further in view of Ota, further in view of Youngdale.

As per claim 14, Flood discloses:

updating said plurality of source-program compile units stored in said object program file so as to be the same as said plurality of source-program compile units constituting said changed source program, updating said plurality of object-program compile units stored in said object program file so as to be the same as said new object-program compile units (Flood)

Flood does not disclose storing analysis information.

However, in an analogous environment, Youngdale teaches:

in addition to storing said plurality of object-program compile units and said plurality of source-program compile units, storing analysis information obtained by syntax analysis of said source program in said object program file (Youngdale bottom of page 2 Listing 1)*;*

updating said analysis information stored in said object program file so as to be the same as the analysis information thus obtained (Youngdale: The format of the ELF file requires that the header information be consistent with the content of the file.).

Youngdale does not expressly disclose comparison of source.

However, in an analogous environment, Ota teaches comparison of two versions source, and compiling the most recent version if they two versions are different (Abstract).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to use Flood's code storage with Ota's comparison method using Youngdale's analysis information . One of ordinary skill would have been motivated to detect changes in code location in an object file which could result in faulty code execution. Checking for consistent object file layout will ensure correct program operation.

As per claim 15, the above rejection of claim 14 is incorporated. The combination of Flood, Caron, and Ota does not expressly disclose storing version information.

However, in an analogous environment, Youngdale teaches the use of a version field in the header of an "ELF" type object file (middle of page 1).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to store Youngdale's version field in Flood's object file. One of ordinary skill would have been motivated to identify the source of an object file. As different compilers use different techniques and algorithms as well as varying optimizations, object debugging is simplified if the source of the object is known.

As per claims 18 and 19, all limitations have been addressed in the above rejections of claims 14, and 15, respectively.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to J. Derek Rutten whose telephone number is (703) 605-5233. The examiner can normally be reached on M-F 6:30-3:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

jdr

[illegible]
TUAN DAM
SUPERVISORY PATENT EXAMINER